Practical 1:-Introduction to R

```
# Lab 1
# Print a message
print('Hello, Welcome to R Programming')
#Assign variables
x <- 25
y <- 5
#Arithmetic operations
sum_val <- x + y
diff_val<- x - y
prod_val <- x * y
div_val<- x / y

sum_val
diff_val
prod_val
div_val
# Relational operations
greater_check <- x > y
equal_check <- x == y
# Load and explore iris dataset
data("iris")  #load dataset
head(iris)    #first 6 rows
str(iris)     #structure of dataset
summary(iris) #descriptive statistics
```

Practical 2:-Importing and Exploring a Dataset in R

```
# Lab 2
#Load required package
library(dplyr)
#1. Import inbuilt dataset: iris
data("iris")
head(iris) #first 6 rows
str(iris) # structure
summary(iris) #summary statistics
#2. Import another dataset: mtcars
data("mtcars")
head (mtcars)
str(mtcars)
#3. Titanic dataset
data("Titanic")
Titanic #view dataset
#4. Check for missing values
sum(is.na(iris)) # total missing values
colSums(is.na(mtcars)) #missing values per column
#5. Summarize distribution by group
iris %>%
  group_by(Species) %>%
  summarise(
    Avg_Sepal_Length = mean(Sepal.Length),
    Avg_Petal_Length = mean(Petal.Length)
  )
```

Practical 3:-Data cleaning and Preprocessing in R

```
# Lab 3
library(dplyr)
# Load dataset
data("airquality")
# Display the first few rows of the original dataset
print("Original Airquality Dataset Head:")
head(airquality)
# 1. Check for missing values
print("--- Missing Value Check (airquality) ---")
# Total NA count across the entire dataset
print(paste("Total NA count:", sum(is.na(airquality))))
# Missing values per column
print("Missing values per column:")
colSums(is.na(airquality))
# 2. Handle missing values
print("--- Handling Missing Values (airquality) ---")
# Replace missing Ozone values with the mean of the column
# 'na.rm = TRUE' ensures the mean is calculated by ignoring existing NA values
airquality$Ozone[is.na(airquality$Ozone)] <- mean(airquality$Ozone, na.rm = TRUE)
# Replace missing Solar.R values with the median of the column
airquality$Solar.R[is.na(airquality$Solar.R)] <- median(airquality$Solar.R, na.rm = TRUE)
# Verify the changes
print("Missing values per column after imputation:")
colSums(is.na(airquality))
# 3. Remove duplicates (using iris dataset as an example)
data("iris")
print("--- Duplicates Removal (iris) ---")
```

```
# Creates a clean dataset by selecting rows that are NOT duplicates.
# 'duplicated()' returns TRUE for the second and subsequent occurrences of duplicate rows.
iris_clean <- iris[!duplicated(iris), ]
# Display the number of rows before and after cleaning
print(paste("Original rows in iris:", nrow(iris)))
print(paste("Rows after removing duplicates:", nrow(iris_clean)))
# 4. Standardize and use formats
print("--- Data Standardization (iris) ---")
# Convert the Species factor to character, then to lowercase for standardization
iris_clean$Species <- tolower(as.character(iris_clean$Species))
# Convert the standardized character column back to a factor
iris_clean$Species <- as.factor(iris_clean$Species)
# View the cleaned dataset head
print("Cleaned Iris Dataset Head:")
head(iris_clean)
```

Practical 4:-Description Statistics and Basic Visualization in R

```
# Lab 4
library(dplyr)
library(ggplot2)
library(modeest) # for mode
# Load dataset
data("iris")
# Descriptive statistics
mean(iris$Sepal.Length) # mean
median(iris$Sepal.Length) # median
mlv(iris$Sepal.Length, method="mfv") # mode
sd(iris$Sepal.Length) # standard deviation
range(iris$Sepal.Length) # min and max
summary(iris$Sepal.Length) # summary
# Histogram
hist(iris$Sepal.Length,
    main="Histogram of Sepal Length",
    xlab="Sepal Length", col="lightblue", border="black")
# Scatterplot
plot(iris$Sepal.Length, iris$Petal.Length,
    main="Scatterplot of Sepal vs Petal Length",
    xlab="Sepal Length", ylab="Petal Length",
    col=c("blue", pch=19))
# Boxplot
boxplot(Sepal.Length ~ Species, data=iris,
        main="Boxplot of Sepal Length by Species",
        xlab="Species", ylab="Sepal Length",
        col=c("lightgreen", "lightblue", "pink"))
```

Prcatical 5:-Variable Transformation and Feature Engineering in R

```
# Lab 5
library(dplyr)
library(caret)
# Load dataset
data("mtcars")
# 1. Binning: Categorise mpg into Low, Medium, High
mtcars$mpg_category = cut(mtcars$mpg,
                          breaks = c(-Inf, 15, 25, Inf),
                          labels = c("Low", "Medium", "High"))
table(mtcars$mpg_category)
# 2. Encoding: Convert Species to numeric codes (iris dataset)
data("iris")
iris$Species_code = as.numeric(as.factor(iris$Species))
head(iris[, c("Species", "Species_code")])
# 3. Normalization: Scale wt (weight) column
mtcars$wt_normalized = (mtcars$wt - min(mtcars$wt)) /
  (max(mtcars$wt) - min(mtcars$wt))
head(mtcars$wt_normalized)
# 4. Standardization: Z-score for hp (horsepower)
mtcars$hp_zscore = scale(mtcars$hp)
head(mtcars$hp_zscore)
# 5. Feature Creation: Power-to-Weight Ratio
mtcars$power_to_weight = mtcars$hp / mtcars$wt
head(mtcars$power_to_weight)
```

Practical 6:-Exploratory Data Analysis (EDA) in R

```
# Lab 6
library(dplyr)
library(ggplot2)
library(GGally)
library(corrplot)
library(ggcorrplot)
# Load dataset
data("iris")
head(iris)
# 1. Summary statistics
summary(iris)
# 2. Histogram of Sepal.Length
ggplot(iris, aes(x = Sepal.Length)) +
  geom_histogram(bins = 15, fill = "lightblue", color = "black") +
  labs(title = "Distribution of Sepal Length", x = "Sepal Length", y = "Frequency")
# 3. Scatterplot Sepal Length vs Petal Length
ggplot(iris, aes(x = Sepal.Length, y = Petal.Length, color = Species)) +
  geom_point(size = 3) +
  labs(title = "Sepal Length vs Petal Length")
# 4. Boxplot of Sepal Width by Species
ggplot(iris, aes(x = Species, y = Sepal.Width, fill = Species)) +
  geom_boxplot() +
  labs(title = "Boxplot of Sepal Width by Species")
# 5. Correlation matrix
corr_matrix = cor(iris[, 1:4])
corr_matrix
ggcorrplot(corr_matrix, lab = TRUE, title="Correlation Matrix Heatmap")
# 6. Pair plot
ggpairs(iris[, 1:4])
```

Practical 7:-Statistical tests In R

```
# Lab 7

library(dplyr)

# Load iris dataset

data("iris")

# 1. t-test: Compare Sepal.Length of setosa and versicolor

t_test_result <- t.test(Sepal.Length ~ Species,

                        data = iris %>% filter(Species %in% c("setosa",
"versicolor")))

t_test_result

# 2. ANOVA: Compare Sepal.Length across all species

anova_model <- aov(Sepal.Length ~ Species, data = iris)

summary(anova_model)

# 3. Correlation: Sepal Length and Petal Length

correlation <- cor(iris$Sepal.Length, iris$Petal.Length)

correlation

# 4. Correlation test with significance

cor_test <- cor.test(iris$Sepal.Length, iris$Petal.Length)

cor_test
```

Practical 8:-Regression Analysis in R

```
# Lab 8
library(dplyr)
library(ggplot2)
# Load dataset
data("mtcars")
head(mtcars)
# 1. Simple Linear Regression: mpg predicted by wt
model_simple = lm(mpg ~ wt, data = mtcars)
summary(model_simple)
# 2. Plot regression line
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point(color = "blue") +
  geom_smooth(method = "lm", se = TRUE, color = "red") +
  labs(title = "Simple Linear Regression: MPG vs Weight",
       x = "Weight (1000 lbs)", y = "Miles Per Gallon")
# 3. Multiple Linear Regression: mpg predicted by wt and hp
model_multiple = lm(mpg ~ wt + hp, data = mtcars)
summary(model_multiple)
# 4. Residual diagnostics
plot(model_multiple, which = 1) # Residuals vs Fitted
plot(model_multiple, which = 2) # Q-Q Plot
```

Practical 9:-Classification Analysis in R

```
library(dplyr)
library(caret)
library(rpart)
library(rpart.plot)
library(ROCR)
library(ggplot2)

data("iris")

iris_bin <- iris %>%
  mutate(Species = factor(ifelse(Species == "setosa", "setosa", "non_setosa"),
                          levels = c("non_setosa", "setosa")))

set.seed(123)
idx <- createDataPartition(iris_bin$Species, p = 0.7, list = FALSE)
trainData <- iris_bin[idx, ]
testData <- iris_bin[-idx, ]

log_model <- glm(Species ~ Sepal.Length + Petal.Length,
                 data = trainData, family = binomial())
log_prob <- predict(log_model, testData, type = "response")
log_class <- ifelse(log_prob > 0.5, "setosa", "non_setosa")

confusionMatrix(factor(log_class, levels = levels(testData$Species)),
                testData$Species)

actual_labels <- ifelse(testData$Species == "setosa", 1, 0)
pred_obj <- prediction(log_prob, actual_labels)
```

```
perf <- performance(pred_obj, "tpr", "fpr")
plot(perf, col = "blue", main = "ROC Curve - Logistic Regression")
abline(a = 0, b = 1, lty = 2, col = "red")

tree_model <- rpart(Species ~ Sepal.Length + Petal.Length,
                    data = trainData, method = "class")
rpart.plot(tree_model)
tree_pred <- predict(tree_model, testData, type = "class")
confusionMatrix(tree_pred, testData$Species)
```

Practical 10:-Clustering Analysis in R(K-Means)

```
# Lab 10
library(dplyr)
library(ggplot2)
library(cluster)
library(factoextra)
# Load dataset (only numeric features)
data("iris")
iris_data <- iris[, 1:4]
# 1. Elbow Method to find optimal K
fviz_nbclust(iris_data, kmeans, method = "wss") +
  labs(title = "Elbow Method for Optimal K")
# 2. Apply K-means clustering with K=3
set.seed(123)
kmeans_model <- kmeans(iris_data, centers = 3, nstart = 20)
# 3. Cluster assignments
kmeans_model$cluster[1:10]
table(kmeans_model$cluster, iris$Species)
# 4. Visualize clusters
fviz_cluster(kmeans_model, data = iris_data,
             ellipse.type = "norm",
             palette = "jco",
             ggtheme = theme_minimal())
```